

FIG. 1

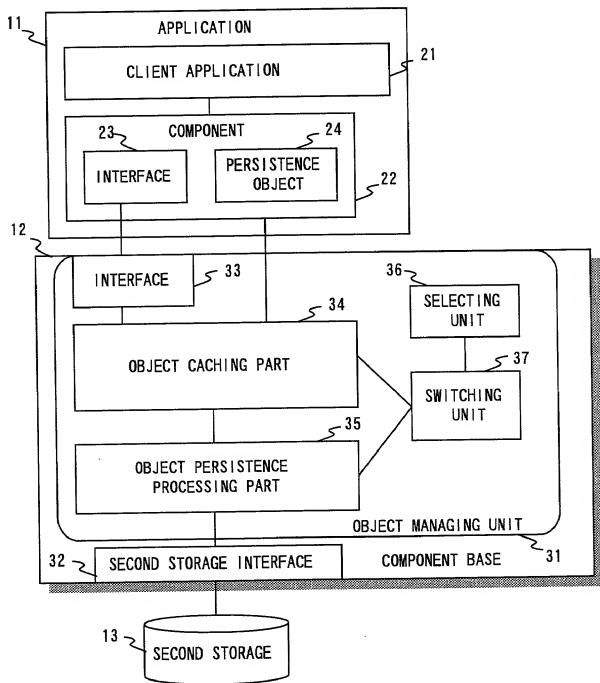


FIG. 2

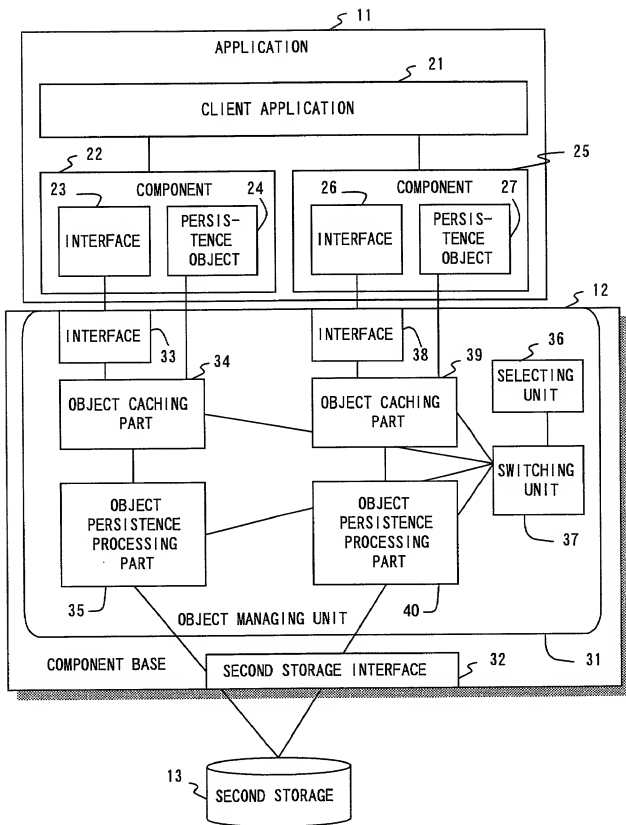


FIG. 3

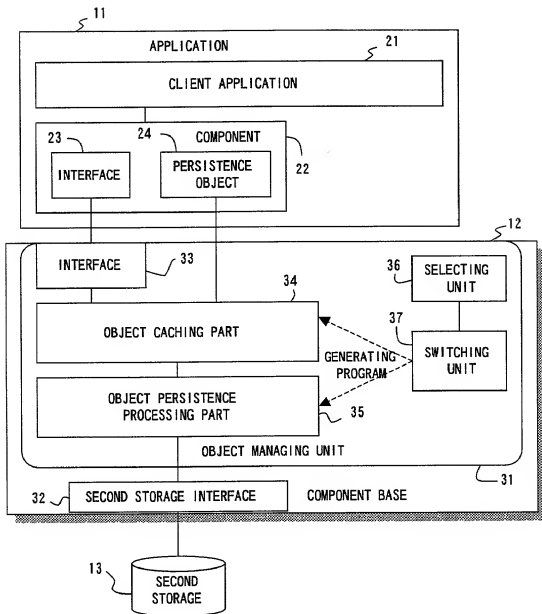


FIG. 4

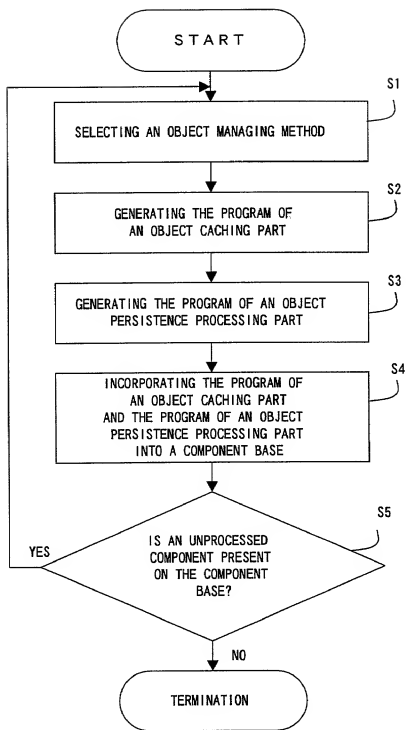


FIG. 5

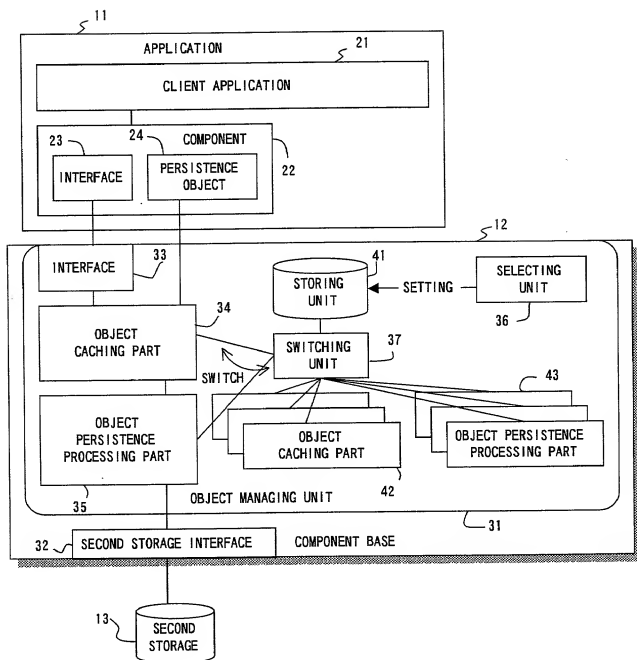


FIG. 6

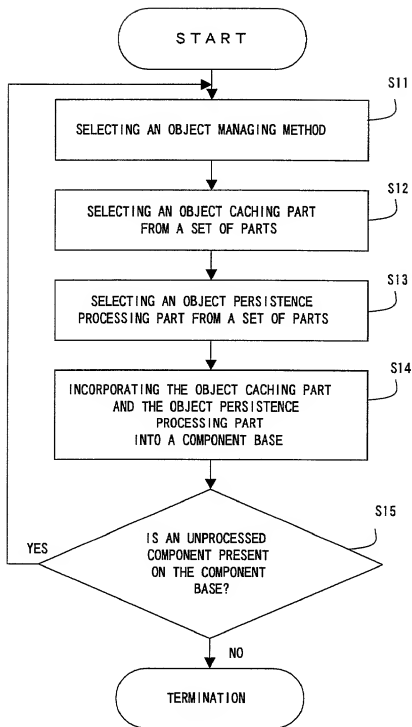


FIG. 7

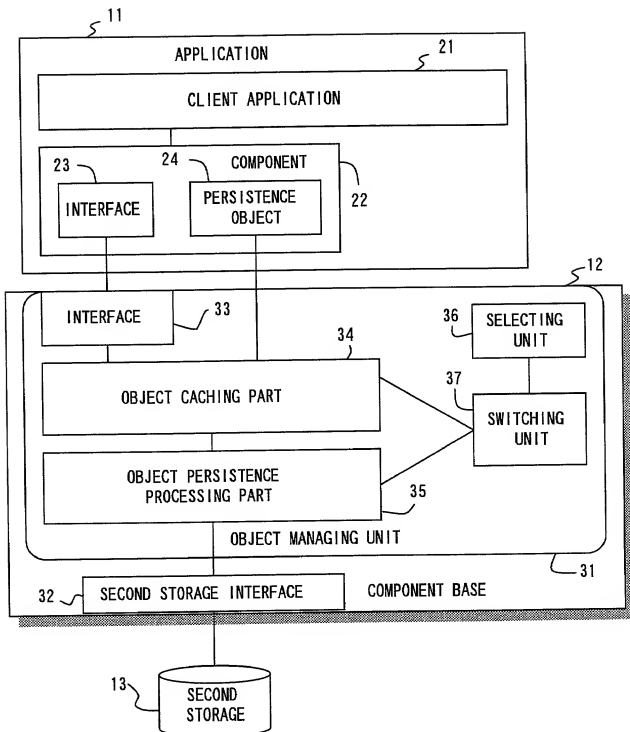


FIG. 8

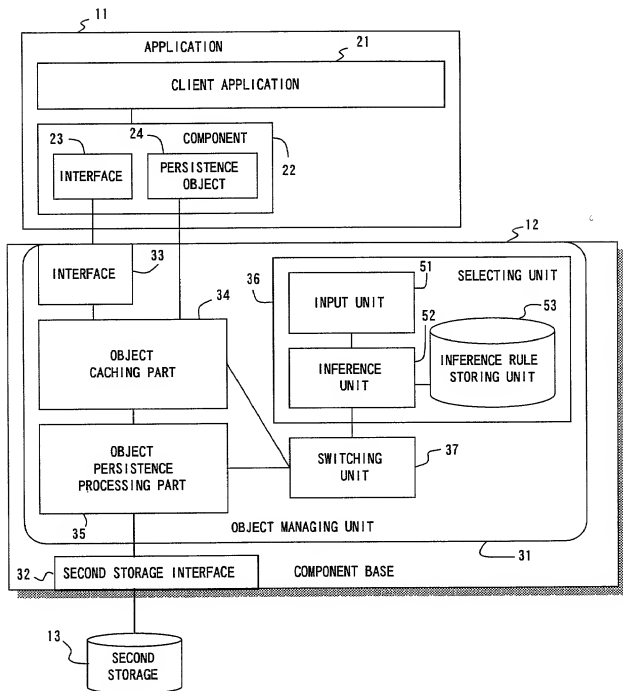


FIG. 9

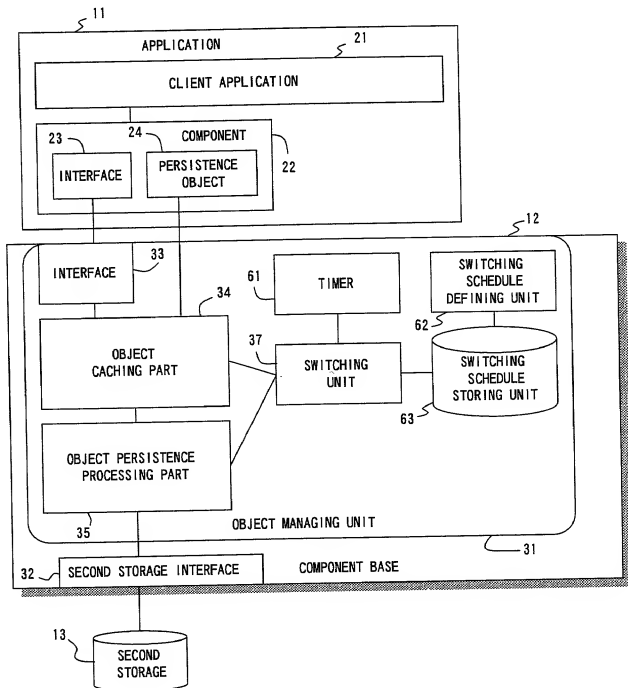


FIG. 10

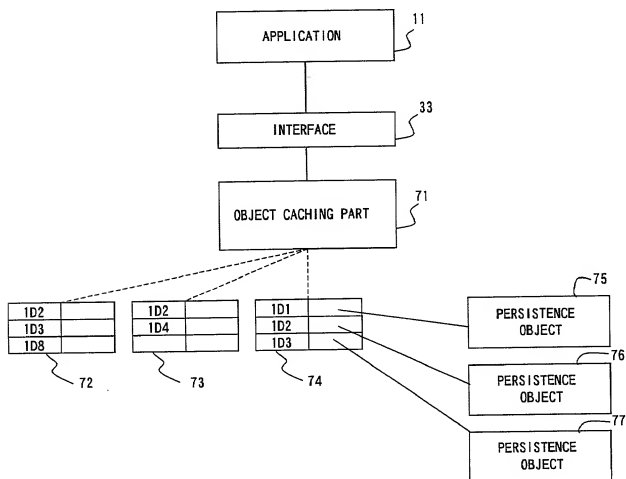
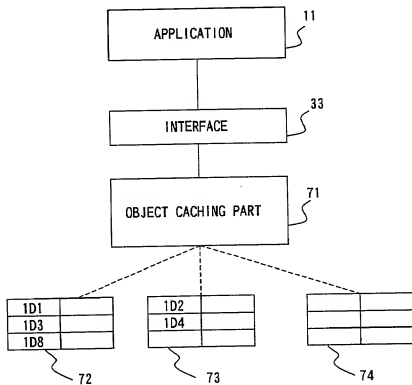


FIG. 11

CONFIDENTIAL



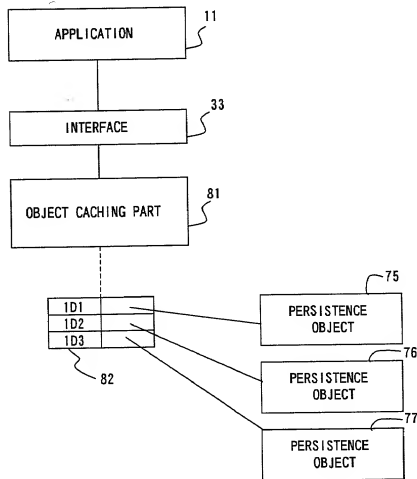


FIG. 13

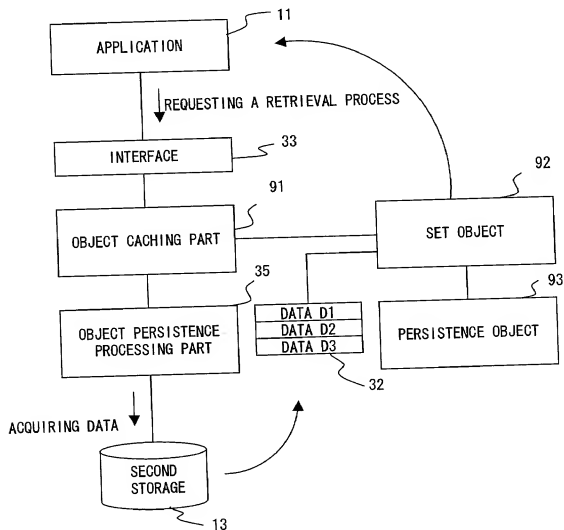


FIG. 14

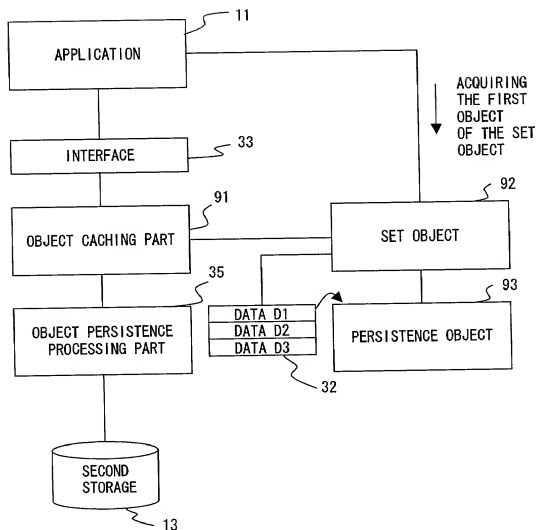


FIG. 15

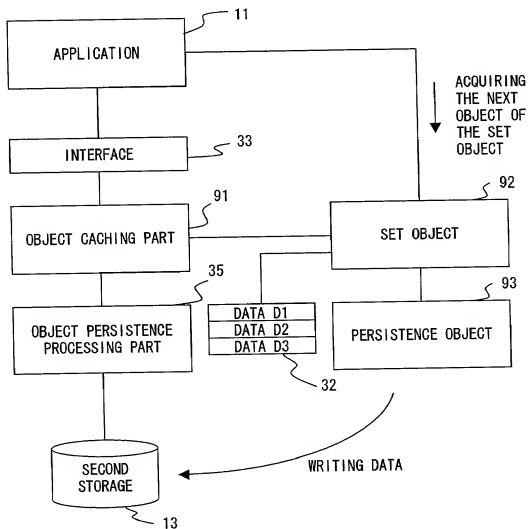


FIG. 16

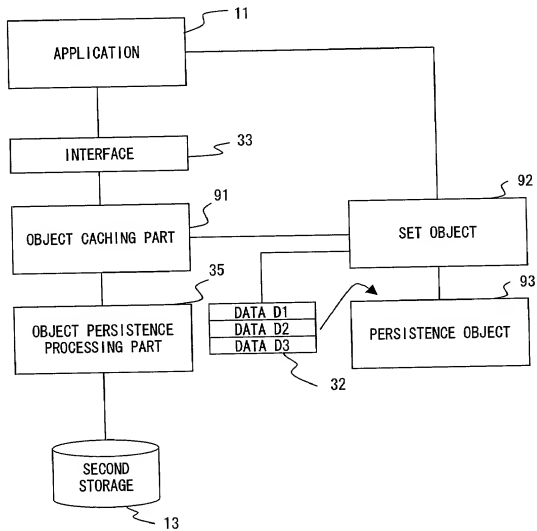


FIG. 17

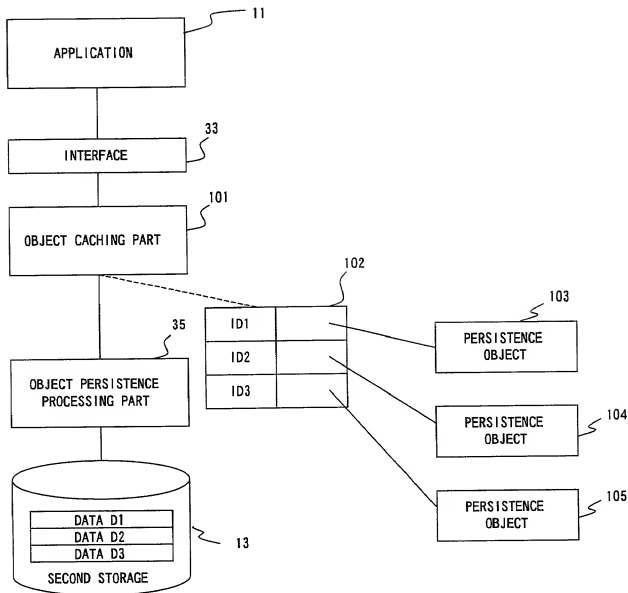


FIG. 18

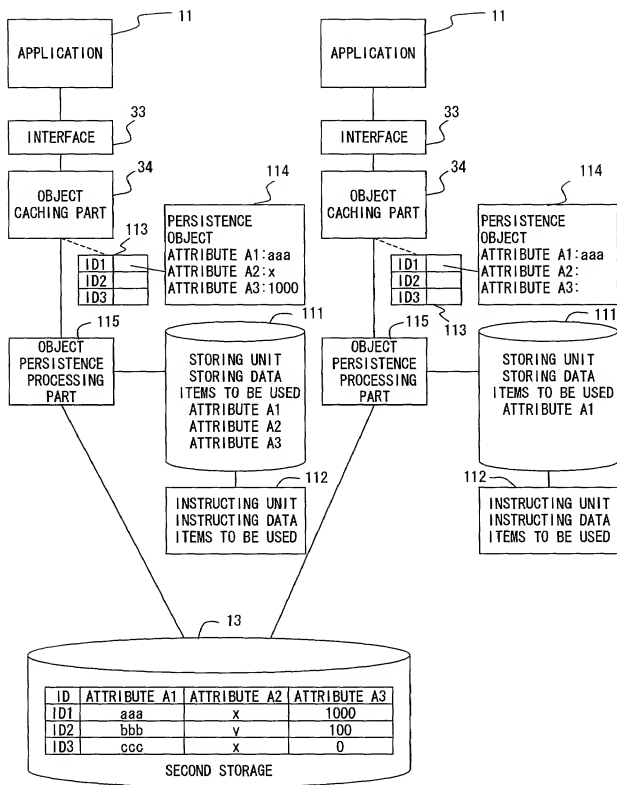


FIG. 19

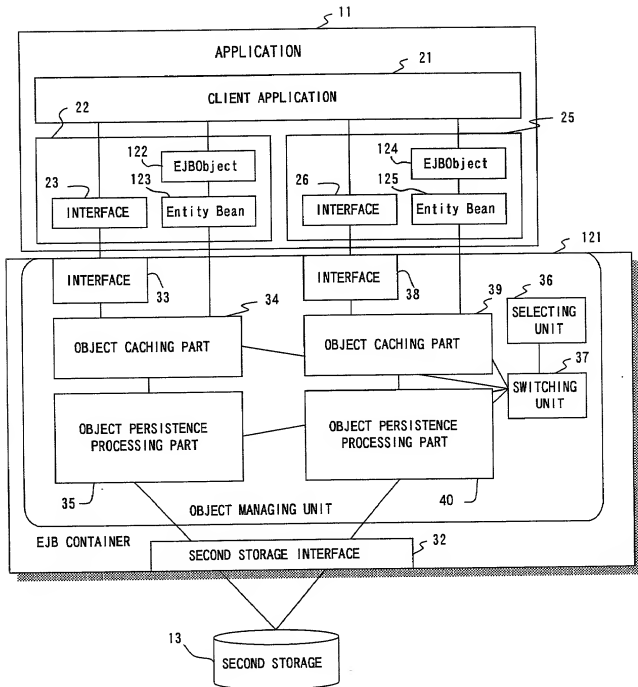


FIG. 20

```
public class $$Bean$$Cache
{
    Hashtable cache$TableHash;
    $$Bean$$Persistence objPersistence;

    $$Bean$$ findByPrimaryKey($$PrimaryKey$$ pk)
    {
        String transactionID = getTransactionID();
        Hashtable cache$Table = (Hashtable)cache$TableHash.get(transactionID);
        $$Bean$$ bean = cache$Table.get(pk);

        if (bean != null) return bean;

        bean = objPersistence.findBean(pk);
        cache$Table.put(pk, bean);
        return bean;
    }
    ....
}
```

FIG. 21

```

public class $$Bean$$Persistence
{
    DataSource dataSource;

    $$Bean$$ findBean($$PrimaryKey$$ pk)
    {
        $$Bean$$ bean = null;
        Connection connection = getConnection();
        String sql = "SELECT $$BeanFieldColumns$$ FROM $$Table$$ WHERE
        $$PKFieldColumns$$ = ?";
        Statement statement = connection.prepareStatement(sql);

        //$$ foreach $$PKFields$$
        statement.set$$Type$$($$Count$$,pk.$$Name$$);
        //$$ end foreach

        ResultSet rs = statement.executeQuery();
        while(rs.next())
        {
            bean = new $$Bean$$();
            //$$ foreach $$BeanFields$$
            bean.$$Name$$ = rs.get$$Type$$($$Count$$);
            //$$ end foreach
        }
        return bean;
    }
    ....
}

```

FIG. 22

```
public class OrderBeanCache
{
    Hashtable cacheTableHash;
    OrderBeanPersistence objPersistence;

    OrderBean findByPrimaryKey(OrderBeanPrimaryKey pk)
    {
        String transactionID = getTransactionID();
        Hashtable cacheTable = (Hashtable)cacheTableHash.get(transactionID);
        OrderBean bean = cacheTable.get(pk);

        if (bean != null) return bean;

        bean = objPersistence.findBean(pk);
        cacheTable.put(pk, bean);
        return bean;
    }
    ....
}
```

FIG. 23

```

public class OrderBeanPersistence
{
    DataSource dataSource;

    OrderBean findBean(OrderBeanPrimarykey pk)
    {
        OrderBean bean = null;
        Connection connection = getConnection();
        String sql = "SELECT ID,PRODUCT,QUANTITY FROM ORDERTABLE
WHERE ID = ?";
        Statement statement = connection.prepareStatement(sql);

        statement.setString(1,pk.id);
        statement.setString(2,pk.product);
        statement.setInt(3,pk.quantity);

        ResultSet rs = statement.executeQuery();
        while(rs.next())
        {
            bean = new OrderBean();
            bean.id = rs.getString(1);
            bean.product = rs.getString(2);
            bean.quantity = rs.getInt(3);
        }
        return bean;
    }
}

```

FIG. 24


```

public class ObjectCacheOption1 extends ObjectCache
{
    Hashtable cacheTableHash;
    ObjectPersistence objPersistences;

    Object findByPrimaryKey(Object pk, BeanDef beanDef)
    {
        String transactionID = getTransactionID();
        Hashtable cacheTable = (Hashtable)cacheTableHash.get(transactionID);
        Object bean = cacheTable.get(pk);
        if (bean != null) return bean;

        bean = objPersistences.findBean(pk, beanDef, "findByPrimaryKey");
        cacheTable.put(pk, bean);
        return bean;
    }
    ....
}

```

FIG. 25

```

public class ObjectPersistenceOption1 extends ObjectPersistence
{
    DataSource dataSource;

    Object findBean(Object pk, BeanDef beanDef, String finderName)
    {
        Object bean = null;
        Connection connection = getConnection();
        String sql = beanDef.getSQL(finderName);
        Statement statement = connection.prepareStatement(sql);

        Enumeration fields = beanDef.getFields();
        while (fields.hasMoreElements())
        {
            FieldDef field = (FieldDef)fields.nextElement();
            setValue(statement,field);

            ResultSet rs = statement.executeQuery();
            while(rs.next())
            {
                bean = beanDef.newInstance();
                fields = beanDef.getFields();
                while (fields.hasMoreElements())
                {
                    FieldDef field = (FieldDef)fields.nextElement();
                    setValue(bean,rs,field);
                }
            }
        }
        return bean;
    }
    ...

    void setValue(Object bean, ResultSet, FieldDef field)
    {
        Field f = field.getField();
        switch (field.fieldType)
        {
            case FT_INT:
                f.setInt(bean,rs.getInt(field.getColumn()));
                break;
            case FT_LONG:
                f.setLong(bean,rs.getLong(field.getColumn()));
                break;
            ...
        }
    }
    ...
}

```

FIG. 26

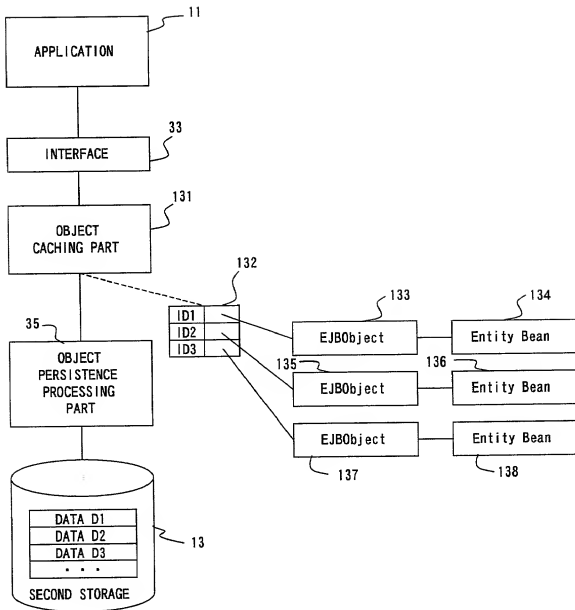


FIG. 27

BUS

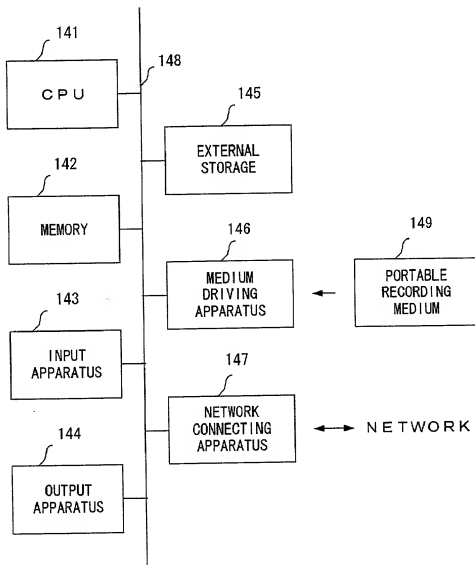


FIG. 28

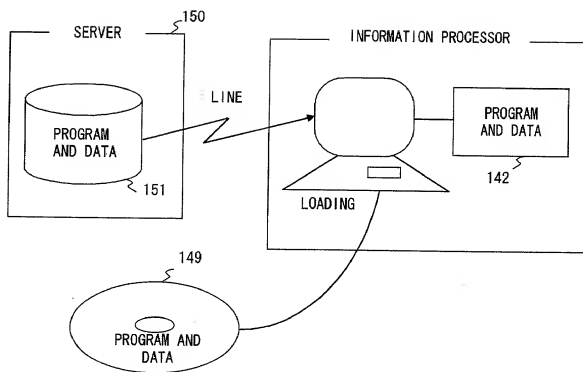


FIG. 29